Software Requirements Specification for RoCam: High Performance Vision-Guided Rocket Tracker

Team #3, SpaceY
Zifan Si
Jianqing Liu
Mike Chen
Xiaotian Lou

October 27, 2025

Contents

Revision History

Date	Developer(s)	Change
Oct. 10, 2025	All team members	Initial release

1 Purpose of the Project

This section explains *why* RoCam exists, *who* benefits, and *how* we will know it succeeds. It states the user need (reliable rocket video at high speed/altitude), the outcome we aim to deliver (autonomous vision-based tracking and recording), and simple success measures that can be verified in the field. It also establishes scope boundaries for early phases (model rockets, short stand-off distances) so that requirements, design, and tests can trace back to clear goals.

1.1 User Business

Student and amateur rocketry teams, launch organizers, and technical judges need reliable, high-fidelity visual evidence of rocket flights to analyze staging behavior, parachute deployment, and flight anomalies that occur at extreme speeds and altitudes. Manual camera operation is impractical in these conditions, and existing tools (manual PTZ/GPS-centric systems) often fail to maintain visual lock on small, fast-moving vehicles. This project addresses this gap with an autonomous, vision-based tracking camera designed to lock onto and follow rockets from launch through landing, providing real-time and recorded footage for post-flight analysis and event operations.

1.2 Goals of the Project

Purpose:

Develop a software+hardware stack (the system) that is able to command a gimbal with a camera mounted on it to track and record model rocket launches.

Advantage:

The system provides insights into the flight performance by having high quality flight footage.

Measure:

Deploy the system at small-scale model rocket launch site with apogee less than 200 meters. The system should provide higher quality flight footage than the manual camera operation.

2 Stakeholders

This section identifies everyone who affects or is affected by RoCam, what they need, and how they will participate. It clarifies responsibilities (client, supervisors, operators, analysts, technicians), priorities, and user characteristics so later requirements and tests are grounded in real roles.

2.1 Client

The client of this project is McMaster Rocketry Team. they are the primary end users who will deploy the system during launches.

2.2 Customer

Apart from our client, we also identified the following customers that may be interested in using the project:

1. **Model Rocketry Event Organizers**: Benefit from reliable tracking for live streaming of rocket flights.
2. **Model Rocketry Safety Officers**: Benefit from live monitoring of parachute deployment and landing location of the rockets.
3. **Aerospace Engineers and Researchers**: Benefit from high-quality flight footage to validate models, improve rocket designs, and support experimental research.

2.3 Other Stakeholders

1. **Dr. Shahin Sirouspour (Supervisor)**: Provides technical guidance, project oversight, and mentorship. Ensures the project aligns with academic standards, engineering best practices, and capstone deliverable expectations.

2.4 Hands-On Users of the Project

Here are the users that will be interacting with the product directly.

Role: Camera Operator

The camera operator controls the system through the UI, starts/stops recordings, manages tracking states, and monitors preview. The camera operator is assumed to have a decent level of familiarity with technology, and have been trained on the use of the product.

Role: Live Stream Technician

The live stream technician manages the video feed from the system to the live streaming equipments. The live stream technician is assumed to have a high level of familiarity with live streaming software and hardware.

Role: Flight Performance Analyst

The flight performance analyst is a member of the rocketry team that reconstructs flight profiles from sensor data and camera footage to understand how the rocket behaves during ascent and descent. The flight performance analyst is assumed to have a high level of familiarity with rocket flight dynamics and software used for analysis.

Role: Live Stream / Recordings Viewers

People who view the live stream or recorded videos. They can be anyone from the general public that are interested in the launch, or the team members themselves. They are assumed to have a basic level of familiarity with technology.

2.5 Personas

Alexis Andrade

Age: 28
Role: Camera Operator

Alexis works as an industrial automation technician in Hamilton, Ontario, where she designs and maintains robotic assembly systems for a manufacturing firm. Outside of work, her real passion lies in high-speed imaging and experimental photography — a hobby that naturally led her to volunteer as a staff camera operator at university rocketry competitions. She loves the mix of engineering precision and adrenaline that comes with capturing a successful rocket launch.

She lives with her partner and their energetic border collie, Pixel. Alexis spends her weekends hiking the Bruce Trail, refurbishing vintage lenses, or volunteering at local maker fairs. Her favorite food is falafel, and she always packs a thermos of strong coffee for early launch mornings. She listens to synthwave and retro electronic music on long drives to test sites, finding it helps her stay calm and focused.

Alexis is confident with technology but prefers systems that are reliable and straightforward. She values hands-on work, hates clunky software interfaces, and believes in simplicity over flashiness. "If I need a manual," she likes to say, "the interface is wrong." Her cool temperament and technical precision make her a trusted member of the launch operations crew.

Brandon Brooks

Age: 23

Role: Flight Performance Analyst

Brandon is in his final year of mechanical engineering at a Canadian university and serves as a flight performance analyst for his rocketry team. His specialty is reconstructing flight profiles from sensor data and camera footage to understand how the rocket behaves during ascent and descent. He joined the team for the engineering challenge but stayed because he loves the mix of theory, experimentation, and teamwork.

Originally from Calgary, Brandon now lives in a small off-campus apartment filled with model rockets, whiteboards, and a curious cat named Fourier. He's known for his patience and meticulous documentation — traits that make him the go-to person for post-launch analysis. When he's not studying or coding MATLAB scripts, he's cooking ramen, taking photos, or biking along Lake Ontario.

Brandon listens to ambient and post-rock while he works, loves strong coffee, and dislikes chaotic testing days where data logging gets overlooked. His attitude toward technology is analytical but practical: he values transparency, well-documented systems, and data that "speaks for itself." For him, every successful flight is a story written in numbers and motion.

Christopher Chou

Age: 59

Role: Event Organizer / Live Stream Technician

Christopher is a retired broadcast engineer who spent three decades working in television production for CBC Toronto. After retirement, he moved to Burlington and found a new passion in volunteer work, helping organize live streams and event logistics for student rocketry competitions. For him, it's the perfect blend of his lifelong love of broadcasting and his fascination with aerospace technology.

He's married with two grown children who live abroad, and he often jokes that the rocketry community has become his "third family." Christopher loves cooking Cantonese comfort food — especially steamed dumplings and congee — and enjoys jazz and classic rock in equal measure. He spends his spare time restoring old audio equipment, gardening, and taking long weekend drives along the Niagara Escarpment.

Christopher's approach to technology is deeply pragmatic: he respects automation but insists that every system should have a manual override. He likes tidy wiring, clear documentation, and software that doesn't crash midstream. He dislikes unnecessary complexity and "updates that fix what wasn't broken." To younger team members, he's both a mentor and the steady hand who keeps the broadcast running when chaos hits.

2.6 Priorities Assigned to Users

Key Users: Camera Operator, Live Stream Technician

Secondary User: Flight Performance Analyst, Live Stream / Recordings Viewers, Event Organizer

2.7 User Participation

Through out the development process, the development team will continue to iterate on the requirements based on feedback from these users:

1. **Camera Operator**: At least three field tests will be conducted to gather feedback from the camera operator. Each field test will last half a day to a day.
2. **Live Stream Technician**: At least two meetings will be conducted to gather feedback from the live stream technician.

2.8 Maintenance Users and Service Technicians

Due to the nature of this project as a capstone requirement, there are currently no expected maintenance users.

3 Mandated Constraints

This section captures external and non-negotiable constraints (budget, schedule, enterprise rules, solution limits). Each constraint includes a short rationale and a fit criterion so compliance is testable and traceable.

3.1 Solution Constraints

SC-1 *The system must use computer vision for rocket detection and tracking*

Rationale: There is already a camera connected to the system for recording, using computer vision for detection and tracking is the most efficient way to do it.

Fit Criterion: The system should be able to detect and track a rocket based only on the camera feed.

3.2 Implementation Environment of the Current System

There are no constrains on the hardware platform the system need to use.

3.3 Partner or Collaborative Applications

The system is expected to be used with a gimbal with a camera mounted on it. The system will command the gimbal to track and record model rocket launches.

3.4 Off-the-Shelf Software

There are no constrains on the off-the-shelf software the system need to use.

3.5 Anticipated Workplace Environment

The system should be designed to operate in a large-scale outdoor launch environment, for example, the Launch Canada launch site in Timmins, Ontario, Or Spaceport America in New Mexico.

The tracking camera will be operated at outdoor launch sites, typically in wide open fields or designated rocket ranges. Launches occur only under clear, sunny conditions with few or no clouds.

Additionally, users often wear hearing protection devices to mitigate high acoustic noise from tools, filling tanks, and generators.

Launch sites are typically in a remote area, with limited or no internet access.

3.6 Schedule Constraints

SHC-1 *The project shall be completed by April 2026, with interim deadlines for key milestones such as Proof of Concept (November 2025) and the final demonstration (March 2026).*

Rationale: These deadlines are based on the academic timeline and the expectations of the capstone course.
Fit Criterion: All project components must be completed and fully functional by the final demonstration in March 2026.
Source PUC: all
Priority: High

3.7 Budget Constraints

BC-1 *The project shall not exceed the budget of 500CAD.*

Rationale: Limited by the capstone project requirement.
Fit Criterion: The sum of the costs of the hardware components required for the project

must be less than 500CAD.
Source PUC: all
Priority: High

3.8 Enterprise Constraints

EC-1 *The product shall be built to comply with the standards of McMaster University's capstone project requirements and academic integrity policies.*

Rationale: The project is part of the university's curriculum and must adhere to its standards.
Fit Criterion: The product must meet the requirements specified by the course syllabus and project advisor.
Source PUC: all
Priority: High

4 Naming Conventions and Terminology

This section defines all terms and acronyms before first use. It provides a single glossary to avoid ambiguity and keep wording consistent across requirements, design, code, and tests.

4.1 Glossary of All Terms, Including Acronyms, Used by Stakeholders involved in the Project

1. **The System**: The software+hardware stack developed for this project that is able to command a gimbal with camera mounted on it to track and record a rocket flight. (the gimbal and the camera are not a part of the system)
2. **Gimbal**: A gimbal is a motorized device that rotates based on the commands sent to it.
3. **Rocket**: Refers to a model rocket.
4. **Ascent**: The stage in a model rocket flight where the rocket is ascending.
5. **Descent**: The stage in a model rocket flight where the rocket has reached its apogee and is descending.

5 Relevant Facts And Assumptions

This section records facts we know and assumptions we rely on (e.g., environment, users, hardware). Making them explicit reduces rework and allows quick updates if conditions change.

5.1 Relevant Facts

None.

5.2 Business Rules

None.

5.3 Assumptions

AS-1 The system will not be connected to the public internet; thus, no authentication is required.

AS-2 The system will only be physically accessed by authorized personnel.

AS-3 Only one rocket will be flying at a time.

AS-4 The gimbal connected to the system has adequate performance to track the rocket (e.g., it can turn fast enough and has enough range of motion).

AS-5 110V AC power is available at the launch site.

AS-6 It will always be sunny at the launch site, because the launch would be cancelled in case of bad weather.

6 The Scope of the Work

This section describes the current real-world work (outside the product), its boundaries, and business events. It frames where RoCam fits and what problems it is trying to solve in the existing process.

6.1 The Current Situation

Currently, the process of recording and analyzing rocket flights relies on manual camera operators positioned at safe distances from the launch pad. These operators attempt to visually follow the rocket during ascent and descent using consumer or semi-professional video cameras mounted on tripods or pan-tilt systems.

This approach suffers from several limitations:

Accuracy: Human reaction time and limited field of view make it nearly impossible to keep fast-moving rockets (often exceeding Mach speeds) centered in frame. The footage is frequently lost during liftoff or staging events, leaving gaps in post-flight analysis.

Safety Constraints: Because a human operator must be physically present, camera placement is restricted to safe zones, which may not provide optimal viewing angles. Cameras cannot always be positioned where the best line-of-sight exists.

Operational Overhead: Each launch requires trained operators, setup time, and coordination with the launch control team. Fatigue, stress, and environmental conditions (sun glare, wind, dust) further degrade performance.

Data Quality: The video produced by manual operators is often shaky, inconsistently framed, and lacking synchronization with other flight data (e.g., telemetry or gimbal angle). This reduces its value for engineering analysis and competition reporting.

6.2 The Context of the Work

Figure 1 identifies the scope of the investigation necessary in order to discover the requirements related to the work of tracking rockets. The model shows the detailed area of investigation and how it connects to the adjacent systems.

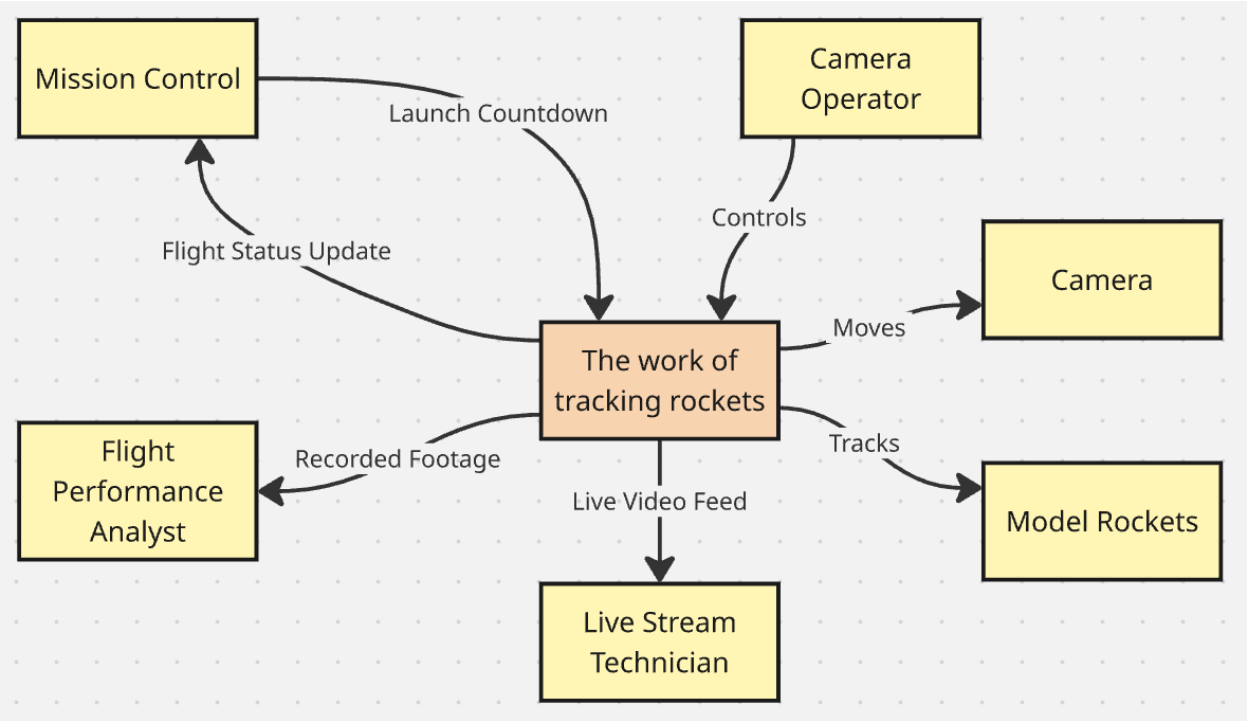

Figure 1: Context of Work Diagram

6.3 Work Partitioning

The Table 1 identifies all business events happening in the real world that affect the work of tracking rockets.

No.	Event Name	Input / Output
BUC 1	Output live video	(in) Video feed from camera (out) Transmitting the video to the live stream
BUC 2	Launch countdown starts	(in) Vocal count down from Mission Control (out) Start recording the rocket flight
BUC 3	Rocket takes flight	(in) Video feed from camera (out) Keeps pointing the camera to the rocket
BUC 4	Rocket completes flight	(in) Video feed from camera (out) Stop tracking the rocket (out) Save the video (out) Update mission control on flight status
BUC 5	Flight performance analyst wants to view the footage	(in) Request from Flight performance analyst (out) Exported video files and logs

Table 1: Work Partitioning of System

6.4 Specifying a Business Use Case (BUC)

Each event listed in Table 1 is expanded into an individual business use case (BUC) which describes business logic in detail.

BUC 1: Output live video.
Trigger: None, always active.
Interested Stakeholders: Model Rocketry Event Organizers
Preconditions: Camera is powered on.
Main Flow of Steps:

1. The camera operator connect the camera to the live streaming equipments.

Outcome: The live stream receives a video feed from the camera.

BUC 2: Launch countdown starts.
Trigger: Mission Control begins the countdown to launch.
Interested Stakeholders: Model Rocketry Event Organizers.
Preconditions: Camera is powered on and ready to record.
Main Flow of Steps:

1. The camera operator points the camera to the rocket

2. The camera operator starts recording.

Outcome: Recording of the rocket flight is started.

BUC 3: Rocket takes flight.
Trigger: Rocket ignition and liftoff are observed.
Interested Stakeholders: Model Rocketry Event Organizers, Aerospace Engineers and Researchers.
Preconditions: Camera is pointing at the rocket and recording.
Main Flow of Steps:

1. The camera operator keeps pointing the camera to the rocket.

2. If the rocket stages, the camera operator points the camera to the next stage.

Outcome: The camera remains pointed at the rocket throughout the flight.

BUC 4: Rocket completes flight.
Trigger: Rocket landing is observed.
Interested Stakeholders: Model Rocketry Event Organizers, Model Rocketry Safety Officers.
Preconditions: Camera is pointing at the rocket and recording.
Main Flow of Steps:

1. The camera operator confirms end of flight based on callout or observation.

2. The camera operator ends the recording.

3. The camera operator stops pointing the camera to the rocket.

4. The camera operator communicates the flight status to Mission Control.

Outcome: Video and logs are safely stored and the flight status is communicated to Mission Control.

BUC 5: Flight performance analyst wants to view the footage.
Trigger: Flight performance analyst requests to view the footage.
Interested Stakeholders: Aerospace Engineers and Researchers.
Preconditions: None.
Main Flow of Steps:

1. The flight performance analyst provides date and time of the flight.

2. The camera operator looks up the flight in the database.

3. If the flight is found, the camera operator exports the video files and logs.
4. If the flight is not found, the camera operator informs the flight performance analyst that the flight is not found.

Outcome: The flight performance analyst receives the video files and logs.

7 Business Data Model and Data Dictionary

This section shows the core business data (launch sessions, footage, logs) and defines each item once. It enables consistent naming, storage planning, and traceability from inputs to outputs.

7.1 Business Data Model

The business data model consists of a list of launches, and for each launch, the video footage and logs associated with the launch.

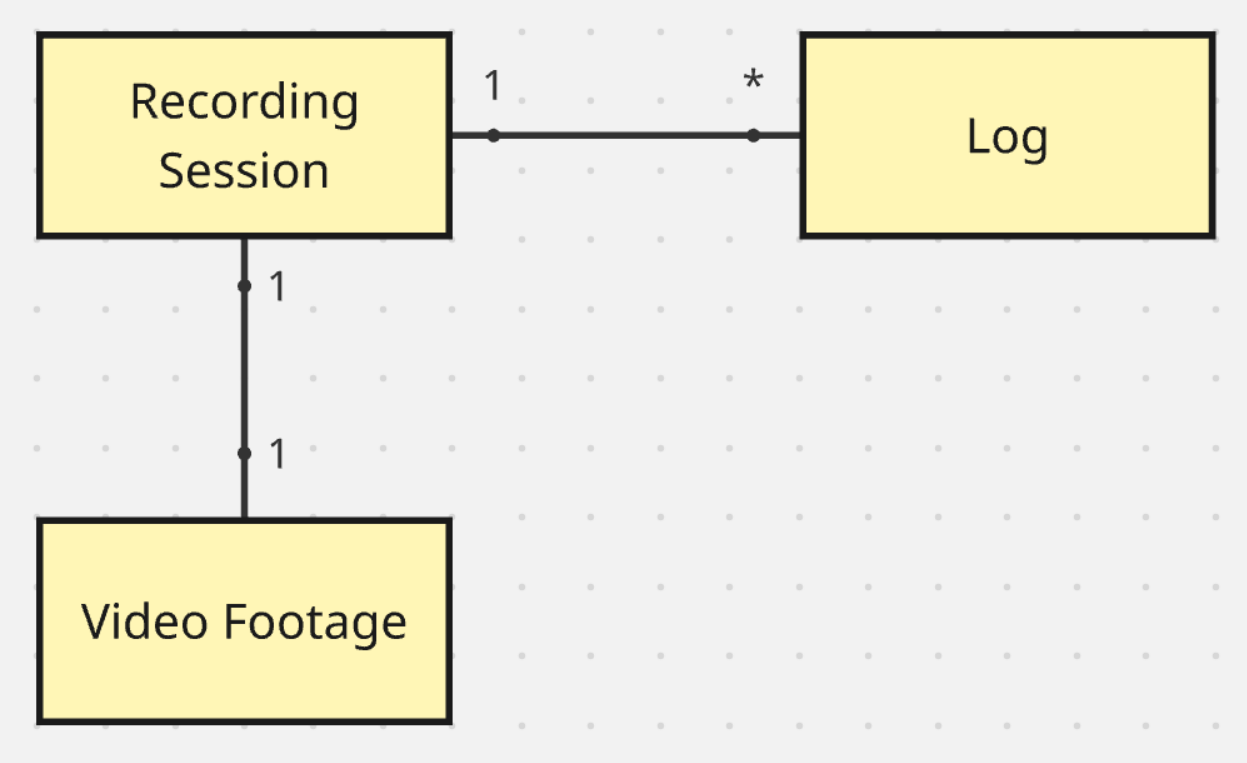

Figure 2: Business Data Model

7.2 Data Dictionary

Name	Content	Type
Recording Session	date + time	Class
Video Footage	Recorded video from the camera	Class
Log	timestamp + gimbal angle	Class

Table 2: Data Dictionary

8 The Scope of the Product

This section draws the boundary of what the product will do, maps BUCs to product use cases, and shows actor interactions. It prevents scope creep and helps readers navigate to the right requirements.

8.1 Product Boundary

The product boundary defines which parts of the work are handled by the system and which parts remain the responsibility of human actors.

In Figure 3, The product use cases (PUCs) are the ellipses inside the boundary. Each PUC is labeled with one or more BUCs that it came from.

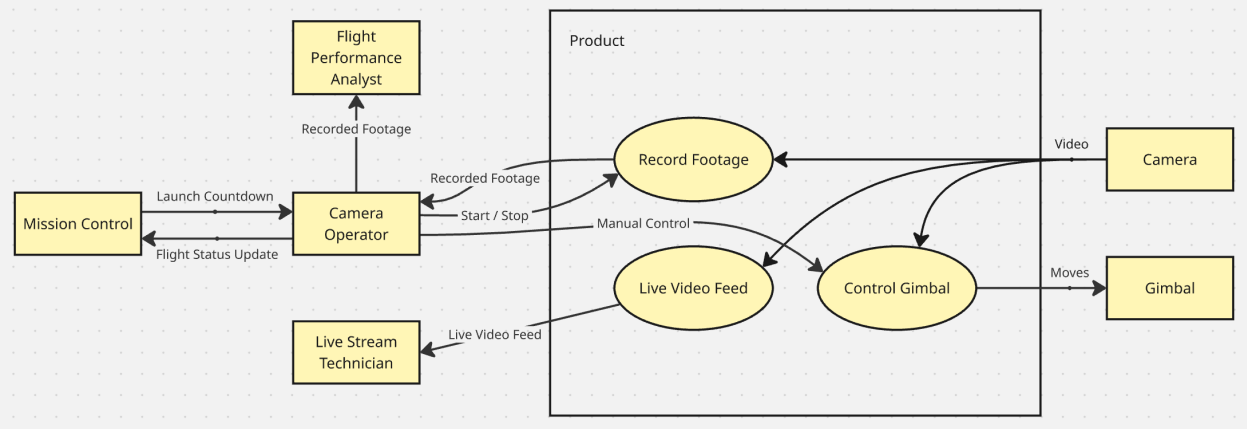

Figure 3: Product Boundary

8.2 Product Use Case Table

The following Table 2 summarizes the primary use cases of the system.

BUC No.	PUC No.	PUC Name	Actors	Input / Output
1	1	Live video output	Camera, Live Stream Technician	(in) Video feed from camera (out) Live video output
2	2.1	Point camera to rocket	Camera Operator, Camera, Gimbal	(in) Manual gimbal adjustment commands from camera operator (out) Movement commands to the gimbal
2	2.2	Start video recording	Camera Operator, Camera	(in) Start video recording command (out) Video recording started
2	2.3	Start tracking rocket	Camera Operator, Camera, Gimbal	(in) Start tracking rocket command (out) System in armed state
3	3	Automated rocket tracking	Camera, Gimbal	(in) Video feed from camera (out) Movement commands to gimbal
4	4.1	Stop video recording	Camera Operator, Camera	(in) Stop video recording command (out) Video recording stopped and saved
4	4.2	Stop tracking rocket	Camera Operator, Camera, Gimbal	(in) Stop tracking rocket command (out) Gimbal stops moving (out) System in idle state
5	5	View recorded footage and logs	Camera Operator	(in) request to download footage and logs (out) Footage and logs downloaded

Table 3: Product Use Case Table

8.3 Individual Product Use Cases (PUC's)

PUC 1: Live video output.
Actors: Camera, Live Stream Technician.
Preconditions: The system is powered on and has finished initialization.

Main Flow of Steps:

1. The camera operator connects the system to the live streaming equipments.

Outcome: The live streaming equipments receive a video feed from the system.

PUC 2.1: Point camera to rocket.
Actors: Camera Operator, Camera, Gimbal.
Preconditions: The system is in idle state.
Main Flow of Steps:

1. The camera operator monitors the camera preview on the UI.
2. The camera operator selects the direction to rotate the gimbal on the UI.
3. The system sends commands to the gimbal to rotate to the selected direction.
4. The camera operator confirms the gimbal is rotating to the selected direction through the preview.

Outcome: The gimbal is rotated to the selected direction.

PUC 2.2: Start video recording.
Actors: Camera Operator, Camera.
Preconditions: The system is powered on and has finished initialization; not currently recording.
Main Flow of Steps:

1. The camera operator clicks the "Start Recording" button on the UI.
2. The system starts to record the camera feed and additional logs (e.g. angle of the gimbal, temperature of the camera, etc.).
3. The UI displays a recording indicator and the elapsed recording time.

Outcome: Video recording starts and logs are written to storage.

PUC 2.3: Start tracking rocket.
Actors: Camera Operator, Camera, Gimbal.
Preconditions: System is in idle state.
Main Flow of Steps:

1. The camera operator selects the "Arm" button on the UI.
2. The system enters the armed state.
3. The UI displays a "Armed" indicator.

Outcome: The system enters the armed state and starts to look for rockets.

PUC 3: Automated rocket tracking.
Actors: Camera, Gimbal.
Preconditions: System is in armed state.
Main Flow of Steps:

1. The system processes incoming frames to detect moving rockets.
2. (Event) The system detects a moving rocket.
3. The system enters the tracking state.
4. The UI displays a "Tracking" indicator.
5. The system sends movement commands to the gimbal to keep the rocket centered in the frame.
6. If the rocket is lost, the system returns to idle state.

Outcome: The gimbal follows the rocket and keeps it near the frame center until the rocket is lost.

PUC 4.1: Stop video recording.
Actors: Camera Operator, Camera.
Preconditions: The system is powered on and has finished initialization; currently recording.
Main Flow of Steps:

1. The camera operator clicks the "Stop Recording" button on the UI.
2. The system stops recording the camera feed and additional logs (e.g. angle of the gimbal, temperature of the camera, etc.).
3. The system saves the video and logs to storage.
4. The UI stops displaying the recording indicator.

Outcome: Video recording stops.

PUC 4.2: Stop tracking rocket.
Actors: Camera Operator, Camera, Gimbal.
Preconditions: System is in armed or tracking state.
Main Flow of Steps:

1. The camera operator clicks the "Stop Tracking" button on the UI.

2. The system returns to idle state.
3. The system stops sending movement commands to the gimbal.
4. The UI displays a "Idle" indicator.

Outcome: The system returns to idle state.

PUC 5: View recorded footage and logs.
Actors: Camera Operator.
Preconditions: The system is in idle state.
Main Flow of Steps:

1. The camera operator opens the recordings page in the UI.
2. The camera operator filters or selects the desired files by date/time.
3. The camera operator downloads the files to their device.

Outcome: The desired footage and logs are downloaded for offline viewing and analysis.

8.4 System State Diagram

The following is a state diagram of the system.

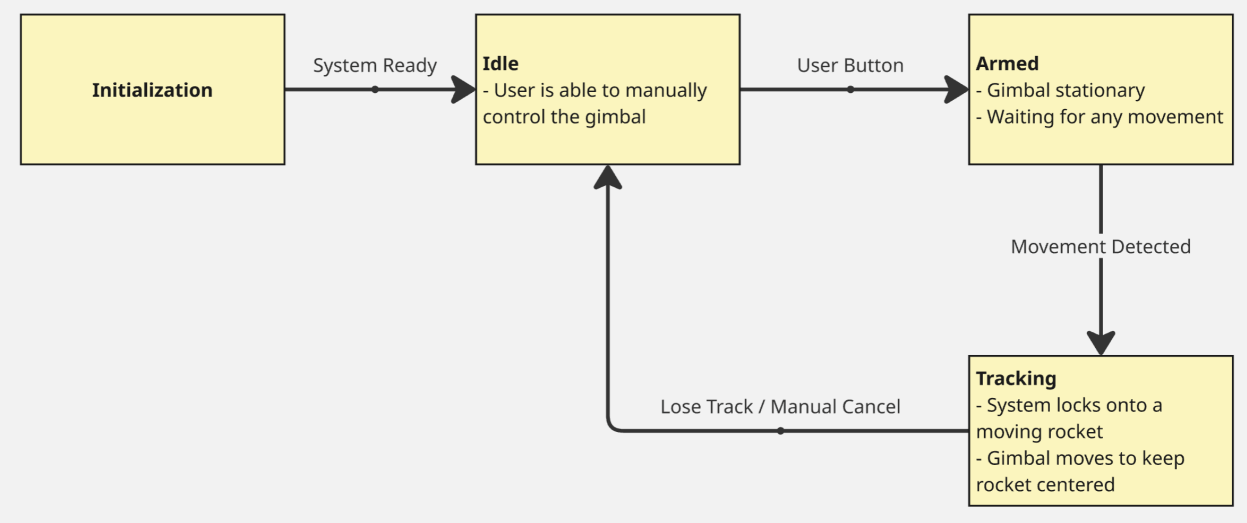

Figure 4: System State Diagram

9 Functional Requirements

This section lists black-box behaviors RoCam must provide. Each FR has a unique ID, rationale, fit criterion, and trace to use cases so it is complete, unambiguous, and verifiable.

9.1 Functional Requirements

FR-1 *The system must acquire live video stream from the connected camera.*

Rationale: Video acquisition enables all downstream processing and recording.
Fit Criterion: The system successfully captures frames from the camera.
Source PUC: 1, 3
Priority: High

FR-2 *The system must send real-time movement commands to the connected gimbal.*

Rationale: Sending movement commands to gimbal is how the system controls the gimbal.
Fit Criterion: Movement commands are sent to the gimbal once per video frame received.
Source PUC: 2.1, 3
Priority: High

FR-3 *The system must detect moving rocket from the video stream in real time.*

Rationale: Real time rocket detection is the first step of the tracking process.
Fit Criterion: The system can detect moving rockets from the video stream in real time.
Source PUC: 3
Priority: High

FR-4 *If the system is in the armed state, the system must automatically enter the tracking state when a moving rocket is detected.*

Rationale: Entering the tracking state means the system is commanding the gimbal to keep the rocket centered in the frame.
Fit Criterion: The system enters the tracking state when a moving rocket is detected.
Source PUC: 2.3, 3
Priority: High

FR-5 *If the system is in the tracking state and the rocket is lost, the system must automatically return to the idle state.*

Rationale: Automatically reverting to idle prevents unnecessary gimbal movement and prepares the system for future launches or manual intervention.
Fit Criterion: Upon loss of the rocket target, the system transitions from tracking to idle state without manual input.
Source PUC: 3
Priority: High

FR-6 *When in the idle state, the system must allow the user to manually control the gimbal.*

Rationale: Manual gimbal control is necessary for arming the camera to the launch pad before the rocket is launched.
Fit Criterion: In idle mode, the user can issue manual pan/tilt commands that are reflected in real-time by the gimbal hardware.

Source PUC: 2.1
Priority: High

FR-7 *When in the tracking state, the system must autonomously keep the rocket within the frame.*

Rationale: Automatic rocket tracking is the core function of the system.
Fit Criterion: Rocket remains within visible in the frame.
Source PUC: 3
Priority: High

FR-8 *The system must begin outputting video through HDMI as soon as it is initialized.*

Rationale: Immediate HDMI output ensures that event video is available to stakeholders from the earliest possible moment.
Fit Criterion: Upon system initialization, an HDMI-connected display immediately receives a live video signal from the camera without requiring additional user action.
Source PUC: 1
Priority: High

FR-9 *The HDMI video output must have a text overlay that displays the current gimbal tilt and pan angle.*

Rationale: Overlaying the gimbal angles on the video stream provides immediate feedback to operators and analysts and aids in diagnostics and post-flight analysis.
Fit Criterion: The HDMI output includes a text label showing current tilt and pan angles, updating once per video frame.
Source PUC: 1
Priority: Medium

FR-10 *The system must display runtime information including system state and camera preview to the user.*

Rationale: Seeing the system status and camera preview is necessary for the user to control the system.
Fit Criterion: The interface updates system status and preview in real time with at least 15fps.
Source PUC: 2.1
Priority: High

FR-11 *The system must allow users to record video and logs associated with the video for later review.*

Rationale: Recording supports debugging, validation, and demonstration.
Fit Criterion: Users can start and stop recording from the UI.
Source PUC: 2.2, 4.1
Priority: High

FR-12 *The system must provide management functionalities for recorded videos and logs (list, delete, download).*

Rationale: Reviewing and managing captured sessions supports performance analysis and documentation.
Fit Criterion: Users can view the list of recorded videos and logs, and perform delete/download actions.
Source PUC: 5
Priority: High

10 Look and Feel Requirements

This section states how the UI should look for outdoor use (readability, layout, professional tone). Criteria are phrased so field checks and usability reviews can confirm conformance.

10.1 Appearance Requirements

AR-1 *The user interface shall be designed for higher readability in outdoor environments*

Rationale: The system is going to be used in outdoor environments
Fit Criterion: Field tests report no issues with readability in outdoor environments
Source PUC: 2.1, 2.2, 2.3, 4.1, 4.2, 5
Priority: High

AR-2 *All UI elements required for tracking operation shall be visible in one page without scrolling.*

Rationale: Operators need immediate awareness of system status during high-pressure launch operations.
Fit Criterion: On a 1920x1080 monitor, all UI elements required for tracking operation are visible without scrolling
Source PUC: 2.1, 2.2, 2.3, 4.1, 4.2, 5
Priority: High

10.2 Style Requirements

SR-1 *The user interface shall appear professional and reliable*

Rationale: The system is controlling a physical gimbal and it is important to gain the trust of the users.
Fit Criterion: The client thinks the system appears professional and reliable
Source PUC: 2.1, 2.2, 2.3, 4.1, 4.2, 5
Priority: Medium

11 Usability and Humanity Requirements

This section sets expectations for ease of use, feedback, learning, accessibility, and internationalization. It provides measurable goals (e.g., response times, success rates) aligned with user needs.

11.1 Ease of Use Requirements

EZ-1 *Units in the user interface shall be consistent.*

Rationale: Consistency reduces user error during high-pressure launch operations.
Fit Criterion: A style audit of UI readouts finds no unit inconsistencies.
Source PUC: 2.1, 2.2, 2.3, 4.1, 4.2, 5
Priority: High

EZ-2 *All user interface interactions shall provide immediate feedback to the user.*

Rationale: Immediate feedback reassures users that the system has registered their actions and helps prevent confusion or repeated actions during critical operations.
Fit Criterion: For all UI actions, a visible confirmation appears within 0.2 seconds of user input in usability tests.
Source PUC: 2.1, 2.2, 2.3, 4.1, 4.2, 5
Priority: High

EZ-3 *The system shall rely on visual indicators for status and alerts.*

Rationale: The work environment may be noisy or require hearing protection, so visual cues ensure users do not miss important updates or warnings.
Fit Criterion: All status and alerts are presented visually in the UI.
Source PUC: 2.1, 2.2, 2.3, 4.1, 4.2, 5
Priority: High

EZ-4 *The product shall help the user to avoid making mistakes.*

Rationale: Because the system is connected to a physical gimbal, a user error might cause significant damage.
Fit Criterion: Every potentially dangerous command presents a confirmation prompt that must be acknowledged before proceeding.
Source PUC: 2.1, 2.2, 2.3, 4.1, 4.2, 5
Priority: High

11.2 Personalization and Internationalization Requirements

PI-1 *The user interface shall support both metric and imperial units*

Rationale: Allowing the user to pick the unit system with which they are most familiar helps reduce user mistakes.
Fit Criterion: Unit preferences can be selected and applied to the UI.
Source PUC: 2.1, 2.2, 2.3, 4.1, 4.2, 5
Priority: Medium

PI-2 *The user interface shall support both English and French languages*

Rationale: Bilingual support meets accessibility and inclusiveness standards for Canadian stakeholders and allows operation by users from different linguistic backgrounds.
Fit Criterion: Users can select either English or French, and all UI text is presented in

the chosen language.
Source PUC: 2.1, 2.2, 2.3, 4.1, 4.2, 5
Priority: Medium

11.3 Learning Requirements

LR-1 *The system shall be easy to learn for new users.*

Rationale: Reducing the learning curve for technically literate users allows for faster and safer adoption, especially in time-pressured launch scenarios.
Fit Criterion: New users with a reasonable levelof technological familiarity who have read the manual are able to successfully perform all product use cases within 20 minutes without external assistance.
Source PUC: 2.1, 2.2, 2.3, 4.1, 4.2, 5
Priority: High

11.4 Understandability and Politeness Requirements

UPR-1 *The system shall use symbols and words that are naturally understandable by the user community.*

Rationale: Using familiar terminology and icons reduces cognitive load and misinterpretation, especially during high-pressure tasks.
Fit Criterion: Client confirms that all UI symbols and labels can be understood without additional explanation.
Source PUC: 2.1, 2.2, 2.3, 4.1, 4.2, 5
Priority: High

UPR-2 *The system should only display information required for normal use cases on the main page (i.e., hide debug information).*

Rationale: Restricting the main interface to essential information keeps the UI clear and reduces distraction or confusion for users.
Fit Criterion: Reviews confirm that only operational and safety-relevant details are visible to users during normal workflows; diagnostic/debug fields require an explicit action to reveal.
Source PUC: 2.1, 2.2, 2.3, 4.1, 4.2, 5
Priority: Medium

UPR-3 *All confirmation dialogs should explain why an action could potentially be dangerous.*

Rationale: Providing context in confirmation dialogs educates users, prevents accidental execution of risky commands, and reinforces safe operational habits when interacting with hardware.
Fit Criterion: All confirmation dialogs for safety-critical actions include a clear, specific explanation of risk, as verified by interface inspection.
Source PUC: 2.1, 2.2, 2.3, 4.1, 4.2, 5
Priority: Medium

11.5 Accessibility Requirements

AR-1 *Status indicators shall remain distinguishable under common color-vision deficiencies.*

Rationale: Ensures legibility for diverse users and conditions.
Fit Criterion: UI passes color-vision simulations.
Source PUC: 2.1, 2.2, 2.3, 4.1, 4.2, 5
Priority: Low

12 Performance Requirements

This section defines measurable non-functional targets: latency, throughput, accuracy, robustness, capacity, safety-critical behaviors. Each item is testable in bench or field conditions.

12.1 Speed and Latency Requirements

SLR-1 *camera-to-gimbal latency shall be low enough to enable continuous rocket tracking.*

Rationale: If the latency is too high, the gimbal will not be able to keep up with the rocket
Fit Criterion: Field tests confirm the latency is low enough to enable continuous rocket tracking.
Source PUC: 3
Priority: High

SLR-2 *The system shall be able to handle 1080p 60fps camera feed*

Rationale: The camera feed is the input to the system
Fit Criterion: The system is able to handle 1080p 60fps camera feed.
Source PUC: 1, 3
Priority: High

SLR-3 *The system shall be able to output 1080p 60fps video via HDMI*

Rationale: The HDMI output is the output of the system
Fit Criterion: The system is able to output 1080p 60fps video via HDMI.
Source PUC: 1
Priority: High

12.2 Safety-Critical Requirements

SCR-1 *The system must allow the user to manually exit armed or tracking mode, transitioning the system to idle mode.*

Rationale: Manual override is necessary to ensure safety, provide user control in unexpected circumstances.
Fit Criterion: When the user issues an exit command during armed or tracking mode,

the system transitions to idle mode within 0.5 second and ceases any automated gimbal movement.
Source PUC: 4.2
Priority: High

SCR-2 *The system must be hands-off (autonomous) from rocket ignition through landing.*

Rationale: During live launches, human operators must prioritize range safety and situational awareness.
Fit Criterion: Once ready, no manual inputs are necessary for the nominal operation of the system until a declared "landed/clear" signal from mission control.
Source PUC: 3
Priority: High

SCR-3 *The system must be remote operatable.*

Rationale: The system may need to be positioned inside hazardous zones (launch pad, ballistic landing areas) or at distant vantage points with no safe operator access during the countdown and flight.
Fit Criterion: The system should be able to perform all the product use cases with no physical interaction at the camera site once deployed.
Source PUC: all
Priority: High

SCR-4 *The system must only enter armed mode when explicitly instructed by the user.*

Rationale: Prevents accidental or unintended transitions to armed state, ensuring safe and predictable operation for all users.
Fit Criterion: The system transitions from idle to armed mode only in response to a clear, user-initiated command, and never automatically or implicitly.
Source PUC: 2.3
Priority: High

12.3 Precision or Accuracy Requirements

PAR-1 *In the recorded footage and the live stream, the rocket shall be centered in the frame.*

Rationale: Stable footage is important for the analysis of the flight
Fit Criterion: The client is satisfied with the stability of the footage.
Source PUC: 1, 3
Priority: High

12.4 Robustness or Fault-Tolerance Requirements

RFR-1 *The system shall report all errors to the user.*

Rationale: Immediate visibility of errors ensures users can respond quickly, improving system safety and reliability.
Fit Criterion: Any error condition, including hardware faults, communication losses,

and software exceptions, generates a visible and descriptive message on the user interface.
Source PUC: all
Priority: High

RFR-2 *The system shall immediately stop all automated operations and enter a safe state if any unrecoverable error occurs.*

Rationale: Rapid cessation of activity prevents damage to equipment and ensures user safety when continued operation is unsafe.
Fit Criterion: Simulated unrecoverable errors cause the system to halt actuation.
Source PUC: all
Priority: High

RFR-3 *The system must operate normally without access to the public internet.*

Rationale: Launch sites are often located in remote areas without reliable internet connectivity.
Fit Criterion: All product use cases are fully functional when the system is operated in an environment with no access to the public internet.
Source PUC: all
Priority: High

RFR-4 *When the system is in the tracking state, and the rocket is not in view, the system must autonomously attempt re-acquisition, before declaring the rocket is lost.*

Rationale: Rocket may be temporarily obstructed, re-acquisition improves the robustness of the tracking process.
Fit Criterion: The system waits for 2 seconds before declaring the rocket is lost.
Source PUC: 3
Priority: Medium

RFR-5 *The system should have a high success rate of tracking the entire rocket flight.*

Rationale: Consistent and reliable tracking throughout the flight is critical for both footage quality and post-flight analysis.
Fit Criterion: The client is satisfied by the success rate during field testing.
Source PUC: 3
Priority: High

12.5 Capacity Requirements

CR-1 *The system must provide sufficient storage capacity to record all launches in one day.*

Rationale: The user may need to keep the video footage until the end of the day.
Fit Criterion: Field tests confirm that the system is able to record and store 100 launches (a reasonable upper bound).
Source PUC: 2.2, 4.1, 5
Priority: High

12.6 Scalability or Extensibility Requirements

None.

12.7 Longevity Requirements

None.

13 Operational and Environmental Requirements

This section describes where and how RoCam operates (temperature, power, networking) and how it connects to adjacent systems (e.g., HDMI). It ensures the design will survive real launch conditions.

13.1 Expected Physical Environment

EPE-1 *The system shall operate normally in the expected temperature range of the deployment environment*

Rationale: Outdoor launch sites can present challenging ambient conditions; hardware and software must tolerate these environments reliably.
Fit Criterion: Test system at 0C to 45C environment temperature and ensure it operates normally.
Source PUC: all
Priority: High

13.2 Wider Environment Requirements

None.

13.3 Requirements for Interfacing with Adjacent Systems

INT-1 *The system must output a live video stream via HDMI.*

Rationale: The customer's live streaming equipments accept HDMI as the integration standard, alternative connectors would require additional adapters.
Fit Criterion: The system should be able to output a live video stream via HDMI, confirmed by connecting a monitor to the system.
Source PUC: 1
Priority: High

INT-2 *The system must provide a flexible interface for gimbal hardware.*

Rationale: Different deployment contexts require different hardware: lighter gimbal for algorithm development and small scale launches, or full-scale gimbal for large scale launches with telescoping lens.
Fit Criterion: A documented gimbal interface for both hardware and software.

Source PUC: 2.1, 3, 4.2
Priority: High

13.4 Productization Requirements

None.

13.5 Release Requirements

RR-1 *One release will be made by the end of the project.*

Rationale: The release schedule is governed by the capstone project timeline.
Fit Criterion: The system is packaged and delivered in a single release at project completion.
Source PUC: all
Priority: High

14 Maintainability and Support Requirements

Not applicable, no features or changes are expected to be made after the project is completed.

14.1 Supportability Requirements

Not applicable.

14.2 Adaptability Requirements

Not applicable, we are free to pick any hardware platform.

15 Security Requirements

This section sets rules for safe control and data integrity in mostly offline operation. It focuses on validating inputs, auditing actions, and minimizing attack surface.

15.1 Access Requirements

Not applicable, we assume that the system will not be connected to the public internet, and only physically accessible by authorized personnel. (AS-1 and AS-2)

15.2 Integrity Requirements

IR-1 *The system shall validate all user inputs to prevent invalid commands from being sent to the gimbal.*

Rationale: Prevents accidental or malicious input errors that could result in unsafe or unintended gimbal motion.
Fit Criterion: All gimbal control inputs from the user interface are checked for validity and range prior to execution.
Source PUC: 2.1, 4.2
Priority: High

15.3 Privacy Requirements

None.

15.4 Audit Requirements

AUR-1 *The system shall log all operations performed by the user, recording each action with a timestamp.*

Rationale: Comprehensive logging increases traceability, supports troubleshooting, and ensures accountability by maintaining an accurate record of user activity.
Fit Criterion: The system logs all operations performed by the user with timestamps.
Source PUC: all
Priority: Medium

15.5 Immunity Requirements

Not applicable, we assume that the system will not be connected to the public internet, and only physically accessible by authorized personnel.

16 Cultural Requirements

This section ensures UI text, symbols, and colours are respectful and broadly acceptable. It commits to simple reviews to avoid insensitive content.

CR-1 *The system must avoid using colours or symbols that could be culturally sensitive or offensive.*

Rationale: Ensuring cultural sensitivity in design helps avoid alienating or offending users from diverse backgrounds, which is critical for global acceptance and usability.
Fit Criterion: Conduct a cultural review to ensure that all icons and colours used in the tool are neutral and universally acceptable.
Source PUC: 2.1, 2.2, 2.3, 4.1, 4.2, 5
Priority: Medium

CR-2 *The system must not include content that could be considered culturally insensitive.*

Rationale: Avoiding culturally insensitive content ensures that the tool is respectful and inclusive, fostering a positive user experience across different cultures.
Fit Criterion: Conduct a cultural review to ensure all content is appropriate for a global

audience.
Source PUC: 2.1, 2.2, 2.3, 4.1, 4.2, 5
Priority: Medium

17 Compliance Requirements

This section lists legal or institutional obligations (privacy, academic policies) and how we will show compliance. Items are phrased with clear acceptance criteria.

17.1 Legal Requirements

LR-1 *The system must respect user privacy by avoiding the collection of any personal or identifiable data.*

Rationale: Ensuring privacy builds user trust and minimizes risks associated with handling sensitive information, even when specific privacy laws are not targeted.
Fit Criterion: The system avoids collecting or storing personal user data, including information about user activities and profiles, and operates entirely within the user's local environment.
Source PUC: all
Priority: High

17.2 Standards Compliance Requirements

None.

18 Open Issues

This section lists open questions that could affect RoCam's design, operation, and deployment. Each issue states why it matters, how we will close it with a test, who owns it, a due date aligned to the POC window (Nov. 17–28, 2025), and trace notes for bi-directional traceability. Acronyms are defined on first use: frames per second (FPS); field of view (FOV).

OI-01 — End-to-end rocket tracking feasibility

Summary: Can we reliably lock, keep centered, and reacquire a small, fast rocket?
Why it matters: This is RoCam's core value.
Closure test: Field tests show a median center error at most 20 pixels and reacquisition in at most 1.0 seconds across at least 5 launches.
Owner: CV Lead; Motion Control Lead **Due:** 2025-11-20
Trace: FR-Tracking-Loop, FR-Reacquire; POC Steps 2–4; V&V Plan.

OI-02 — Jetson pipeline performance (capture to inference to control)

Summary: Validate FPS and latency on target hardware.
Why it matters: Low latency enables smooth gimbal motion.
Closure test: End-to-end latency at most 120 ms while sustaining at least 30 FPS; report median and 95th percentile.
Owner: CV Lead **Due:** 2025-11-10
Trace: NFR-Performance; FR-Tracking-Loop; Expected Technology (CV).

OI-03 — Gimbal slew and STM32 control stability

Summary: Check max pan/tilt speeds and controller stability under rocket rates.
Why it matters: Hardware must keep up without oscillation.
Closure test: Step and ramp tests show no instability; max slew at least 180 deg/s; overshoot at most 5 percent.
Owner: Motion Control Lead **Due:** 2025-11-10
Trace: FR-Gimbal-Control; NFR-Safety; POC Step 4.

OI-04 — Occlusion, smoke, glare and backlight handling

Summary: Robust tracking in difficult visuals; automatic reacquire.
Why it matters: Launches include plume/smoke and changing light.
Closure test: Scripted tests with occluders and backlight achieve at least 90 percent successful reacquisition within 1.0 seconds.
Owner: CV Lead; QA **Due:** 2025-11-15
Trace: FR-Reacquire, FR-Tracking-Loop; V&V scenarios.

OI-05 — Web preview latency and control responsiveness

Summary: Measure browser preview delay and UI command round-trip on field network.
Why it matters: Operator needs near real-time feedback.
Closure test: Preview 95th percentile latency at most 300 ms; command round-trip at most 150 ms on site Wi-Fi or hotspot.
Owner: Web Lead **Due:** 2025-11-15
Trace: FR-Web-Preview, FR-Remote-Control; NFR-Usability.

OI-06 — Target size versus lens, FOV, and stand-off distance

Summary: Ensure rocket has enough pixels for detection and tracking.
Why it matters: Too small a target breaks detection.
Closure test: Optics sizing shows at least 30 pixels of target height at the minimum distance; confirmed in ground tests.

Owner: CV Lead; HW Lead **Due:** 2025-11-05
Trace: FR-Image-Acquisition, FR-Detect-Motion; POC Step 1.

OI-07 — Recording and archival (format, bitrate, storage, retrieval)

Summary: Decide how to record video and make it available.
Why it matters: Needed for demos, analysis, and V&V evidence.
Closure test: Recording at 1920x1080 and 30 FPS with storage rotation; operator can download footage via the web UI.
Owner: Web Lead; CV Lead **Due:** 2025-11-18
Trace: FR-Recording, FR-Web-Download; NFR-Storage.

OI-08 — Site networking and power plan

Summary: Confirm bandwidth and stable power at launch site(s).
Why it matters: Unreliable links or power will break the demo.
Closure test: Site checklist complete; backup link tested; UPS or batteries sized for at least 30 minutes of runtime.
Owner: Field Ops; HW Lead **Due:** 2025-11-12
Trace: NFR-Reliability; Ops-Readiness; POC logistics.

OI-09 — Safety and abort in the web UI

Summary: Define operator authority, safe states, and emergency stop flow.
Why it matters: Must halt motion safely on faults or command.
Closure test: From any state, emergency stop reaches a safe state within at most 200 ms; procedure documented and tested.
Owner: Motion Control Lead; Web Lead; QA **Due:** 2025-11-16
Trace: NFR-Safety; FR-Gimbal-Control; Hazard Analysis.

OI-10 — Integration interfaces (CV, STM32, Web)

Summary: Freeze message schemas and timing expectations.
Why it matters: Reduces late integration risk.
Closure test: Versioned API documents published; bench integration test passes three consecutive runs without timeouts.
Owner: Subproject Leads; QA facilitator **Due:** 2025-11-08
Trace: Cross-cutting FRs; POC risk “Integration”.

OI-11 — Environmental readiness (wind, lighting, Jetson thermals)

Summary: Validate operation under expected weather and heat.
Why it matters: Outdoor launches vary.
Closure test: Test matrix executed; Jetson temperatures at or below 80 C with no throttling during a 30 minute run.

Owner: Platform Eng.; Field Ops **Due:** 2025-11-14
Trace: NFR-Performance; NFR-Reliability; Ops-Readiness.

OI-12 — Operator workflow and usability

Summary: Make target pick, mode switch, and errors simple and clear.
Why it matters: Reduces operator mistakes during demos.
Closure test: Five-user hallway test: at least four of five complete tasks unaided; System Usability Scale score at least 75.
Owner: Web Lead; QA **Due:** 2025-11-18
Trace: NFR-Usability; FR-Target-Selection; Web App.

OI-13 — Install and update (Jetson/STM32) and rollback

Summary: Define install steps, versioning, and rollback.
Why it matters: The rubric requires installability; reliable updates reduce field risk.
Closure test: Cold install in at most 10 minutes; one-command update and rollback verified on a clean device.
Owner: Platform Eng. **Due:** 2025-11-12
Trace: NFR-Installability; CI/CD; Expected Technology.

OI-14 — Data retention and privacy

Summary: Set retention period and access rules for footage and logs.
Why it matters: Manages storage and sharing expectations.
Closure test: Policy documented; maximum retention of 30 days enforced automatically; access control tested.
Owner: PM; Web Lead **Due:** 2025-11-18
Trace: NFR-Security and Privacy; FR-Recording and Logs.

19 Off-the-Shelf Solutions

19.1 Ready-Made Products

Conclusion: No off-the-shelf product reliably tracks a small, fast rocket end-to-end.

Surveyed: consumer/prosumer camera trackers; sports/wildlife auto-tracking; industrial/defense PTZ trackers.

Criteria: acquire/re-acquire small target; keep centered at high speed; low latency; operator control/safety; integration (camera, gimbal, web); logging.

Findings (Gaps):

- Target too small / motion too fast.
- Weak occlusion/smoke re-acquire.
- Latency not specified or too high.
- Closed stacks; poor API/integration.

- Limited safety controls and logs.

Decision: Build custom solution (meets performance, safety, integration needs).

Revisit: Reassess if a vendor demonstrates low-latency, small-target tracking with open APIs.

19.2 Reusable Components

Below are the key open-source components RoCam will reuse. For each, we explain what it does, where it comes from, and why it is a good fit. Each choice is simple to verify with clear measurements and has low integration risk.

OpenCV (Python). OpenCV will handle camera capture, reading and writing frames, basic image operations, and drawing overlays for the live preview. It is open source under a BSD license and is widely used on NVIDIA Jetson devices. We can verify it easily by measuring frames per second and end-to-end latency during live capture and by checking that overlays appear correctly on the preview.

Ultralytics YOLO. Ultralytics YOLO will provide object detection to find the rocket and keep the track initialized. It is open source and has models that balance accuracy and speed on edge hardware. We will test it on sample launch videos and measure precision, recall, and inference time to confirm it meets our target speed and detection quality.

PyTorch and TorchVision. PyTorch and TorchVision will run the detection models, perform preprocessing, and support simple experiments such as quantization. They are open source with a permissive license and work well with CUDA on the Jetson platform. We will verify them by timing the full pipeline on our device and confirming that model outputs are correct on a small labeled clip set.

Optical Flow (OpenCV). We will use OpenCV's optical flow to add motion cues that help with small or fast targets and to support re-acquisition if the detector misses a frame. This component is open source and lightweight. We will compare re-acquisition time and tracking continuity with and without optical flow on the same sequences to confirm the benefit.

NVIDIA JetPack SDK. JetPack provides CUDA, cuDNN, multimedia libraries, and device drivers required for the Jetson. It is a free vendor SDK. Using JetPack reduces custom integration work and enables hardware acceleration. We will verify correct setup by running a small GPU self-test, confirming camera capture works, and measuring that model inference uses the GPU.

STM32 with `embassy-rs` (Rust). The STM32 microcontroller and the `embassy-rs` async runtime will run the real-time gimbal control loop, including timers, GPIO, and serial links. These crates are open source and build on `embedded-hal`. We will verify control

quality using simple step and ramp tests, checking for stable response, acceptable overshoot, and maximum slew rate.

Flask (Backend). Flask will serve a small HTTP and WebSocket API for control, status, and video preview endpoints. It is open source under a BSD license, easy to containerize, and simple to read and test. We will verify the API with unit tests for each route and with a basic integration test that starts, arms, begins tracking, and stops through the web interface.

React (Frontend). React will power the operator web interface for connect, arm, start and stop tracking, target selection, and status. It is open source under an MIT license and has a strong ecosystem. We will verify usability with short hallway tests, looking at task completion, time to perform key actions, and a simple usability score.

GStreamer or FFmpeg. We will use GStreamer or FFmpeg for video encoding, streaming, and recording. Both are open source and proven on Jetson. We will verify preview and recording by measuring median and tail preview latency, checking for dropped frames, and confirming that recorded files play back at the expected resolution and frame rate.

Docker. Docker will be used to containerize the computer vision and backend services to ensure repeatable builds and installs. It is open source (Moby/Apache 2.0). We will verify installability by performing a clean install on a fresh device, running one command to start services, and testing a simple rollback of a container image.

GitHub Actions. GitHub Actions will provide continuous integration for builds, unit tests, style checks, and container images. It has a free tier for public use. We will verify it by enforcing that each pull request runs linting and tests, produces a build artifact, and fails if coverage or style gates are not met.

Verification approach for all components. For every component above we will record the version and license, define one or more clear checks (for example frames per second, latency, precision and recall, controller step response, and web task success), and store the results with links back to the requirement they support. This keeps our reuse choices measurable and traceable.

19.3 Products That Can Be Copied

RoCam can adopt proven interface and workflow patterns from adjacent products to reduce risk and speed delivery. The most relevant sources are: *surveillance PTZ camera consoles* (clear live preview, crosshair, presets), *drone ground control stations* (arm/disarm, mode switching, prominent safety actions), *IP/action camera apps* (record/stop, timers, storage indicators, simple download), and *low-latency web viewers* (connection status, latency badges, reconnect prompts). These systems solve problems similar to ours: showing live video with minimal delay, keeping critical controls obvious, and helping operators recover from errors quickly.

Adaptation potential. For RoCam, we will copy the following ideas: (1) a single primary live view with a fixed crosshair and status banner; (2) large, consistent controls for *Start Tracking*, *Stop*, *Arm*, and *Abort*; (3) quick target selection directly on the preview; (4) a visible latency/FPS indicator and connection health pill; (5) simple recording controls with a clip timer and remaining storage; (6) a basic "Downloads" page for recorded clips and logs.

Why copy: These patterns are familiar to operators, lower training time, and reduce mistakes under pressure. Reuse also helps us meet usability and safety requirements with less custom design.

What not to copy: We will avoid vendor-specific jargon, buried expert menus for critical actions, and multi-step wizards for emergency functions. Abort/Stop must remain always visible and one-click.

Verification: Each adopted pattern will have a small, testable check in the V&V plan (for example: *max two clicks to start tracking*, $\leq$ *1 second to stop gimbal motion*, $\geq$ *80% of users complete target selection unaided*, *operator SUS score meets the usability threshold*). Results will be recorded and traced to the corresponding usability and safety requirements.

Considerations: Ensure accessibility (clear labels, contrast, keyboard navigation), align terminology with the project glossary, and keep links to the related requirements for traceability. Only copy ideas and layouts—do not copy proprietary code or branding. Check licenses for any third-party assets before use.

Documentation for each copied idea: For every pattern we adopt, we will record: name and source, purpose in RoCam, the requirement(s) it supports, a brief integration note, any licensing notes, and its verification method and status.

20 New Problems

This section highlights impacts, risks, and user challenges introduced by RoCam and how we will mitigate them. It keeps long-term sustainability in view beyond the first demo.

20.1 Effects on the Current Environment

RoCam is designed to operate without negative effects on the current environment. It will run on university-owned hardware in the lab and at outdoor test sites. Our goal is to integrate safely with existing systems and to conduct field tests without disrupting other users or activities.

In the campus and lab setting, the main risks are extra load on shared compute resources, additional storage for video and logs, and higher network traffic from live preview and uploads. To avoid problems, we will set conservative defaults for frame rate and bitrate, monitor resource use during early runs, and respect storage and bandwidth quotas agreed with the infrastructure team.

At field sites, the concerns are physical safety, power and heat, privacy, and local wireless interference. We will secure the gimbal and tripod, provide an emergency stop that puts the system in a safe state, manage cables to prevent trip hazards, and size batteries so that power is stable. We will avoid recording people or property by accident by announcing tests

and positioning the camera away from bystanders. If wireless links are used, we will choose channels that do not interfere with local networks.

We will verify that there is no negative impact by measuring CPU/GPU, storage, and network usage on a campus dry run and confirming they stay within agreed limits. In the field, we will use a short checklist to confirm emergency stop behavior, cable safety, and battery capacity before each session. For privacy, we will review a sample recording to ensure no unintended personal data is captured and confirm that access to downloads requires authentication. Evidence of these checks (resource screenshots, checklist sign-offs, and retention settings) will be included in the V&V package.

20.2 Effects on the Installed Systems

RoCam connects to the existing live stream setup through an HDMI cable from the Jetson video output to the venue's switcher or capture device. HDMI is an industry standard, so no negative effects on the installed systems are expected. The system will operate as a passive video source: it does not send control signals to other devices, does not modify mixer settings, and does not require driver installs on the production hardware.

To align with the rubric's focus on clarity and constraints, we document the following assumptions and limits. The receiving device accepts the agreed video format (for example, 1080p at 30 frames per second) and standard color range. The HDMI cable length and quality meet vendor guidance, and the venue power is stable. RoCam will not draw power or control from the downstream equipment; it uses its own power supply. These constraints reduce integration risk and keep responsibilities clear.

Potential risks and mitigations are simple and testable. If the receiver does not support the default resolution, we will switch to a compatible mode from a short, preapproved list (for example, 720p at 60 frames per second). If signal integrity issues appear (dropouts or flicker), we will replace the cable, add an inline repeater, or reduce resolution to increase margin. If the venue prefers SDI, we will use an HDMI-to-SDI converter that is powered and rated for the selected format. None of these mitigations require changes to the venue's equipment configuration.

Verification is straightforward and recorded. Before events, we will run a five-minute signal check: confirm stable video on the mixer's multiview, verify audio is either muted or at the expected level, and note the detected input format on the mixer UI. We will capture a photo or screenshot of the mixer status, note the cable used and resolution, and store this evidence with the test log. Passing these checks confirms that RoCam does not adversely affect the installed systems and meets the nonfunctional requirement for compatibility.

Traceability is maintained by linking this subsection to the nonfunctional compatibility requirement (NFR-Compatibility), the field readiness checklist, and the proof-of-concept demo plan. Any change to the output format or cabling will be reflected in the checklist and the interface specification so that documentation stays consistent and easy to navigate.

20.3 Potential User Problems

Potential user challenges for **RoCam** include operator onboarding, learning the tracking workflow, and confusion about UI states (IDLE, TRACKING, SEARCH, LOST), manual

override, and recording/preview behavior. Live stream technicians may also struggle to judge when the feed is stable enough to go on-air, especially after a target loss or during high motion.

Motivation: Ensure field operators and live stream technicians can adopt RoCam quickly, run safe and smooth sessions, and correctly interpret status, controls, and video outputs without frustration or missteps.

Considerations:

- Provide a role-based quick start: *Operator* (connect, arm, select target, recover from loss) and *Technician* (preview latency guidance, when to cut/return to live, recording checks).
- Add a short setup wizard: camera connect, lens/FOV check, gimbal zero, network test, and health checks (FPS, latency).
- Include an in-UI legend and tooltips for all states and controls (IDLE, TRACKING, SEARCH, LOST; Arm, Start/Stop, E-stop, Manual Mode).
- Keep recovery simple: one-click *Reselect Target*, clear *Lost* banner, and a brief inline "What to do now" hint.
- Offer a *Practice Mode* using recorded clips so users can rehearse target selection, loss, and reacquire without hardware.
- Provide a one-page *Field Cheat Sheet* (pre-flight checklist, common issues, recovery steps, contact).
- Ensure safety affordances: visible E-stop, confirmation for risky actions, and persistent indication when Manual Mode is active.
- Make the preview easier to interpret: show measured FPS and latency; warn when latency exceeds a threshold.
- Support accessibility (high-contrast theme, larger fonts) for outdoor glare and small screens.
- Establish a support path: issue template for logs/screen captures, contact channel, and a known-good configuration to roll back to.

20.4 Follow-Up Problems

Potential long-term issues for **RoCam** include sustaining the system after the capstone, keeping models and firmware current, and maintaining reliable field operations as hardware and environments change.

Motivation: Ensure RoCam remains safe, usable, and effective beyond the project timeline, with clear ownership and a simple path to updates and repairs.

Considerations:

- **Ownership & Maintenance:** Define who owns the Jetson, STM32 gimbal controller, and web service after handover; name a maintainer for bug triage and releases.
- **Updates:** Plan for JetPack (CUDA/cuDNN), OS security patches, Flask/React dependencies, and STM32 firmware updates; keep a versioned changelog.
- **Model & Tuning Drift:** Re-test detector/tracker performance when models or thresholds change; keep a small benchmark clip set and publish FPS/latency/accuracy results.
- **Calibration & Wear:** Schedule lens focus/FOV checks, gimbal zeroing, and mechanical inspection (slop/backlash); store a simple calibration checklist.
- **Spare Parts & Repairs:** Keep spare cables, power supplies, SD cards, and a backup camera; document swap procedures.
- **Data & Storage:** Manage recording retention and log rotation; verify backup/restore of configs and calibration files.
- **Docs & Training:** Maintain a one-page field cheat sheet, a quick-start guide, and short training clips; refresh operator training each term.
- **Safety & Compliance:** Preserve E-stop tests, safe-state procedures, and hazard checks; review site permissions and flight-day rules yearly.
- **Support Path:** Provide a contact channel, an issue template (with logs/versions), and a known-good image to roll back to.
- **CI/CD & Release Hygiene:** Keep CI for lint/tests/builds; tag releases, publish binaries/containers, and attach V&V summaries for traceability.

21 Tasks

This section explains how we will deliver RoCam in phases with clear milestones, demos, and feedback loops. Each phase links work to requirements and V&V to maintain traceability.

21.1 Project Planning

RoCam will be delivered in short Agile iterations that fit within the capstone milestones. Each iteration includes planning, build, code review, and a testable demo slice (camera capture $\rightarrow$ tracking $\rightarrow$ gimbal control $\rightarrow$ web preview). We use GitHub for issues, pull requests, and CI, VS Code/PyCharm for development, and LaTeX for documents. Work happens on the edge (NVIDIA Jetson with an STM32 controller) with a browser-based operator UI. Non-functional items (safety, installability, usability, field readiness) are planned alongside features and tracked as requirements.

Initiation (Problem, Scope, Risks, Success). We confirm the problem and scope, list stakeholders (operator, field ops, supervisor), identify key risks, and define success criteria. We outline the proof-of-concept steps and agree on a hardware-in-the-loop approach. *Deliverables:* Problem Statement, POC Plan, Development Plan.

Requirements (What, not How). We capture functional requirements (camera input, detection, tracking, gimbal control, web preview, recording) and non-functional requirements (performance, safety, usability, installability, reliability). We write the SRS and Hazard Analysis and define measurable KPIs such as FPS, end-to-end latency, center error, and reacquire time. *Deliverables:* SRS, Hazard Analysis.

Design (Architecture and Interfaces). We define the system architecture, data flows, and interface contracts between Jetson, STM32, and the Web App. We specify message schemas, UI flows (connect, arm, start/stop, select target), and the streaming/recording path. We choose optics and placements based on field of view and minimum target size. *Deliverables:* Design Document (initial, then revision 0).

Implementation (Build and Integrate). We implement camera capture, motion/detection and tracking, gimbal control on STM32, the operator UI, and video preview/recording. We add CI checks (lint, unit tests, build), and containerize the CV/backend on Jetson. The goal is a working end-to-end slice that can be demonstrated safely. *Deliverable:* Proof-of-Concept demo.

Validation and Verification (Measure and Prove). We write tests for modules and for integrated flows. On the bench, we measure controller step response and maximum slew. In the field, we test occlusion, glare, and reacquire behavior. We run basic usability checks (task success and simple surveys). All tests trace back to requirements and KPIs. *Deliverables:* V&V Plan and V&V Report.

Deployment and Field Operations (Package and Demonstrate). We prepare install, update, and rollback scripts; write an operations runbook and safety procedures; and complete a site checklist for network and power. We conduct the final demo and hand over documentation and instructions for ongoing use. *Deliverables:* Final Demo, Final Documentation, Operations Guide.

Iteration Cadence and Tools. We plan two short iterations per milestone with a review at the end of each. GitHub Projects tracks issues and priorities; GitHub Actions runs CI; Docker images pin runtime dependencies; simple checklists guard field tests. Every change is linked to a requirement to keep traceability clear and to support fast onboarding of new contributors.

21.2 Planning of the Development Phases

Insert your content here.

21.3 Planning of the Development Phases

RoCam will be delivered in short iterations that align with the capstone milestones. Each phase below states the benefit to users, the main components exercised (Jetson CV pipeline, STM32 motion control, Web App), and the key functional (FR) and non-functional (NFR) requirements addressed. Feedback from the supervisor and field operators is gathered at the end of every iteration and feeds the next one.

Requirements & Analysis (to 2025-10-06). *Benefit:* Clear and testable "what, not how." *Components:* GitHub Issues/Projects, LaTeX docs. *Focus:* Define FRs for camera input, detection/tracking, gimbal control, web preview/controls, recording; define NFRs for performance (FPS, latency), safety (abort/estop), installability, and usability. Output: SRS (Rev 0) and Hazard Analysis.

Architecture & Design (2025-10-07 to 2025-11-10, then Rev 0 by 2026-01-19). *Benefit:* Stable interfaces reduce integration risk. *Components:* Jetson–STM32–Web API contracts, stream/record path, optics choices. *Focus:* Message schemas, timing budgets, UI flows (connect/arm/start/stop/target pick), FOV/target-size envelope. Output: Design doc (draft by 2025-11-10, Revision 0 by 2026-01-19).

Implementation (POC build 2025-09 to 2025-11; continued through 2026-02). *Benefit:* Usable slices early (see-track-move). *Components:* Camera capture, YOLO/flow tracking, STM32 control loops (`embassy-rs`), Flask backend, React operator UI, GStreamer/FFmpeg for preview/recording, Docker, GitHub Actions. *Focus:* Deliver end-to-end path (image → track → control → preview). Output: Proof-of-Concept Demo (2025-11-17 to 2025-11-28).

Validation & Verification (plan 2025-10-27; execute through 2026-03). *Benefit:* Measured confidence against KPIs. *Components:* Bench rigs, recorded clips, field tests, CI test suites. *Focus:* Verify latency/FPS, center error, reacquire time, max slew and stability, occlusion/glare handling, UI task success and command round-trip. Outputs: V&V Plan (2025-10-27), Revision 0 Demo (2026-02-02 to 2026-02-13), V&V Report (2026-03-09).

Deployment & Field Operations (2026-03). *Benefit:* Safe, repeatable operation in the field. *Components:* Install/update/rollback scripts, runbook, site checklist (network/power), safety procedures. *Focus:* Cold-install time, update/rollback, thermal checks, go/no-go criteria, operator training. Outputs: Final Demonstration (2026-03-23 to 2026-03-29).

Handover & Documentation (by 2026-04-06). *Benefit:* Maintainable system after the course ends. *Components:* Operations guide, API references, interface versioning, known-issues list. *Focus:* Traceability from requirements to tests, onboarding guide for new team members, maintenance plan and ownership. Output: Final Documentation (Revision 1, 2026-04-06).

Traceability and Feedback Loops. Each feature and test references a unique requirement ID in GitHub Issues and the SRS. Phase gates occur at the milestone dates above; review notes and measurements are logged and linked to the affected FRs/NFRs to keep bi-directional traceability and ensure continuous alignment with stakeholder expectations.

22 Costs

To meet a hard cap of **$250 CAD** for RoCam, we will **reuse existing lab/Personal equipment** wherever possible and purchase only small, critical items. Labour cost is $0 (capstone). All software is open source.

Assumptions (In–Kind / Existing)

- **Compute:** NVIDIA Jetson dev kit (lab) — $0.
- **Camera & storage:** USB/CSI camera (1080p) and microSD/SSD (lab) — $0.
- **Mounting/power/network:** Tripod, PSU, basic cabling, campus Wi-Fi (lab) — $0.
- **Gimbal:** Use an existing pan–tilt unit if available; otherwise see fallback below.
- **Tooling:** University GitHub, documentation tools, and 3D printer access — $0.

Planned One-Time Purchases (Prototype)

- STM32 dev board (e.g., Nucleo/"Blue Pill") **$25**
- Servo driver (PCA9685) & level shifter **$18**
- Wiring/connectors (Dupont kit, headers, heat-shrink) **$15**
- Mounting plate & fasteners (camera/gimbal to tripod) **$20**
- Small protective enclosure for controller **$20**
- E-stop mushroom switch (inline power cut) **$15**
- Adhesives/ties/misc. consumables **$12**

Subtotal (base case, gimbal reused): **$125**
Contingency (15%): **$19**
Estimated total (base case): $144

Fallback if No Lab Gimbal

- Budget pan–tilt kit or two metal-gear servos **$35**
- (Optional) 3D-print filament for brackets if needed **$10**

Subtotal with fallback added: **$170**
Contingency (15% on full spend): **$26**
Estimated total (with fallback): $196 (*still* $\leq$ $250)

Operational (Per Demo Day)

- Field hotspot day pass (only if site Wi-Fi is unavailable) ..... **$5–$10**
- Transport/consumables ..... **$0–$10** (minimize via campus locations)

Rubric Alignment Notes

- **Clarity & traceability:** Each cost is tied to a system function (control, safety, mounting).
- **Verifiable:** All line items are commodity parts with public pricing; contingency is explicit.
- **Phase-in:** Base case meets POC under $250; optional gimbal fallback also stays under cap.

23 User Documentation and Training

This section defines what must be documented for **RoCam** and how operators will be trained. All items include clear ownership and measurable acceptance criteria to align with the rubric's verifiability and traceability goals.

23.1 User Documentation Requirements

23.1.1 User Manual (Operator Guide)

Scope: Install, setup, normal operation, and recovery.

- **Content:** hardware setup (camera, gimbal, power), wiring checks, software start/stop, web app usage (connect, arm, start/stop tracking, select target), indicators and states, common errors and recovery steps, safety and abort procedures.
- **Format:** PDF in `docs/` and a browsable `README` section in the repo vision-guided-tracker.
- **Owner:** Development team; QA is responsible for review.

- **Update policy:** Update with every release that changes UI, workflow, or safety procedure.
- **Acceptance:** A new user can complete a standard session (power on, connect, track, stop, power down) following the manual only; task success rate $\geq 4/5$ users in a hallway test.

23.1.2 Quick Start Card (1–2 pages)

Scope: Field checklist for time-critical steps.

- **Content:** pre-flight checklist, start sequence, target selection, abort sequence, shutdown.
- **Placement:** `docs/quickstart.pdf` and printed sheet with the kit.
- **Acceptance:** An operator can set up and begin tracking in ≤ 5 minutes using only the card.

23.1.3 Operator Safety Checklist

Scope: Mandatory safety steps and go/no-go criteria.

- **Content:** site and weather checks, people and equipment safe distances, e-stop verification, network fallback, thermal check on Jetson.
- **Acceptance:** Checklist completed and signed for every field session; incident-free operation in dry runs.

23.1.4 API and Interface Reference

Scope: Web API endpoints and STM32 control messages.

- **Content:** versioned message schemas, timing expectations, error codes, sample requests.
- **Acceptance:** Integration test passes using only this reference; schemas are versioned and linked from the Design and V&V docs.

23.1.5 Maintenance and Troubleshooting Guide

Scope: Logs, common faults, part replacement, calibration.

- **Content:** how to collect logs, reset procedures, gimbal calibration steps, spare parts list.
- **Acceptance:** A non-developer can recover from the top 5 known faults using this guide in ≤ 10 minutes per fault.

23.1.6 Release Manual

Scope: Build, package, deploy, rollback.

- **Content:** CI steps (GitHub Actions), image build, versioning scheme, Jetson install script, STM32 flashing steps, rollback procedure, release notes template.
- **Owner:** Development team; Platform engineer approves.
- **Acceptance:** A clean device can be installed from scratch in ≤ 30 minutes; rollback completes in ≤ 5 minutes.

23.2 Training Requirements

Audience and format. Primary users are the camera operator and live stream technician. Training consists of a 30–45 minute hands-on session plus the Quick Start Card for field use.

Learning objectives (measurable).

- **LO-1: Basic operation.** After training, users can power up, connect, arm, start tracking, select a target, and stop tracking without help. *Criterion:* $\geq 90\%$ task success across 5 users.
- **LO-2: Status understanding.** Users can interpret tracking states and FPS and respond to "lost" by reselecting a target. *Criterion:* $\geq 4/5$ correct responses in scenario questions.
- **LO-3: Safety and abort.** Users can trigger e-stop and recover to a safe state. *Criterion:* Complete e-stop drill in ≤ 10 seconds, two consecutive trials.
- **LO-4: Basic troubleshooting.** Users can follow the guide to resolve a connection or preview issue. *Criterion:* Resolve a seeded issue in ≤ 10 minutes using the guide only.

Materials and upkeep.

- **Materials:** User Manual, Quick Start Card, Safety Checklist, short tutorial video (optional), practice dataset (recorded clip).
- **Upkeep:** Documents updated at every release; tutorial video re-recorded only when UI or safety flow changes.
- **Ownership:** Development team maintains content; supervisor coordinates training sessions for new users.

Evaluation and sign-off.

- Short post-training checklist and a 5-question quiz on states and safety.
- System Usability Scale (SUS) quick survey; *target:* score ≥ 70.
- Training completion is recorded; operators are cleared for field sessions upon passing LO-1 to LO-4 criteria.

24 Waiting Room

This section lists potential features for **RoCam** that are not required for the initial Proof-of-Concept (POC) or Revision 0, but may be added later. Each item includes a brief rationale, a measurable fit criterion, and a priority to support planning and rubric alignment.

WTRM 1. *Auto-zoom during tracking.*

Rationale: Keep the rocket at a consistent screen size to aid visibility.
Fit Criterion: With auto-zoom on, the rocket's height remains within $\pm 15\%$ of a target pixel height for at least 80% of frames during a test clip.
Priority: Medium

WTRM 2. *Multi-target detection with operator target swap.*

Rationale: Handle multiple moving objects (e.g., chaff, birds) and let the operator switch targets.
Fit Criterion: System renders unique IDs for ≥ 3 simultaneous detections and switches lock to a selected ID within ≤ 1 s.
Priority: Low

WTRM 3. *Predictive guidance using simple trajectory estimation.*

Rationale: Reduce control lag when the rocket accelerates quickly.
Fit Criterion: With prediction on, mean center error (pixels) is reduced by $\geq 15\%$ on benchmark clips versus baseline.
Priority: Medium

WTRM 4. *IMU/compass fusion for gimbal stabilization.*

Rationale: Improve stability in wind or operator bump.
Fit Criterion: With fusion enabled, peak pointing deviation under a 5° step disturbance is reduced by $\geq 20\%$ compared to vision-only.
Priority: Low

WTRM 5. *Multi-camera handoff (fixed wide + tracking narrow).*

Rationale: Use a wide FOV to reacquire when the narrow FOV loses the rocket.
Fit Criterion: When the narrow view loses the target, the system reacquires from the wide view in ≤ 2 s in 4/5 test runs.
Priority: Low

WTRM 6. *Night / low-light mode (IR or high-gain pipeline).*

Rationale: Extend operating hours and lighting conditions.
Fit Criterion: Tracking remains stable at 10 lux scenes with FPS $\geq$ target_fps-5 and no more than 10% drop in precision on test clips.
Priority: Low

WTRM 7. *On-device recording with bookmark/annotation in UI.*

Rationale: Easier review of key flight moments.
Fit Criterion: Operator can set at least 5 bookmarks per session; exported MP4 contains embedded timecodes; bookmarks listed in a CSV.
Priority: Medium

WTRM 8. *Health monitoring dashboard (thermal, FPS, packet loss).*

Rationale: Detect performance issues early in the field.
Fit Criterion: UI shows live Jetson temperature, tracking FPS, and preview latency; generates a warning when any exceeds set thresholds for >10 s.
Priority: Medium

WTRM 9. *Operator profiles and saved presets (FOV, gains, limits).*

Rationale: Reduce setup time and errors across sessions.
Fit Criterion: Operator can save and load named presets; loading applies within $\leq$ 1 s; presets persisted across reboots.
Priority: Low

WTRM 10. *Weather and site checklist integration in the web UI.*

Rationale: Improve safety and go/no-go decisions.
Fit Criterion: Pre-flight checklist must be acknowledged before "Arm"; checklist stored with session logs and can be exported as PDF.
Priority: Medium

WTRM 11. *Remote viewer role (read-only web preview share).*

Rationale: Let stakeholders watch without control privileges.
Fit Criterion: A read-only session link streams preview at $\geq$ 720p and p95 latency $\leq$ 400 ms on test network; no control endpoints exposed.
Priority: Low

WTRM 12. *Automatic calibration helper (center, limits, dead-zone).*

Rationale: Faster field setup and consistent control feel.
Fit Criterion: Wizard completes in $\leq$ 3 minutes and stores updated limits; post-calibration step response meets current acceptance thresholds.
Priority: Low

25 Ideas for Solution

This section lists simple, high-level ideas we can explore for **RoCam**. These are not final designs. They help us pick practical options during the proof of concept and later phases.

- **Detect then track.** Start with a fast object detector to find the rocket, then use a light tracker to follow it between detections. *How we check it:* measure frames per second and how close the target stays to the center.
- **Use motion cues.** Add basic motion detection to highlight moving objects and help the detector when the rocket is small. *How we check it:* compare miss rate with and without motion cues on the same video.
- **Plan for lost target.** If lock is lost, search near the last known spot and allow the operator to reselect the target quickly. *How we check it:* time to reacquire and percent of successful reacquisitions.
- **Simple, stable control.** Run a basic controller for pan and tilt on the microcontroller, with limits on speed and acceleration. *How we check it:* bench tests for step and ramp moves, looking for no oscillation.
- **Small, clear messages.** Send compact messages between the Jetson and the microcontroller with a checksum and a counter. *How we check it:* log packet loss and late packets during a long run.
- **Straightforward video path.** Start with an easy preview method first (for example, MJPEG) and record video locally; only move to lower-latency streaming if needed. *How we check it:* measure preview delay and dropped frames; confirm recorded files play back.
- **Simple web UI.** Give the operator a clean page with connect, arm, start, stop, target pick, state display, and an emergency stop. *How we check it:* quick user tests for task completion and command round-trip time.
- **Fast setup and calibration.** Provide a short routine to find center offset, limits, and pixel to degree scale; save and reuse settings. *How we check it:* after calibration, the system reaches targets without large error or overshoot.
- **Pick the right lens.** Estimate how many pixels tall the rocket will be at expected distances to guide lens and camera choice. *How we check it:* compare predicted and measured pixel sizes on test targets.
- **Protect performance and thermals.** Watch device temperature and reduce non-essential work if limits are near; show warnings in the UI. *How we check it:* run for 30 minutes without throttling; alarms fire at set limits.
- **One place for settings.** Keep important thresholds in a single constants file and allow safe changes through the UI. *How we check it:* no hidden magic numbers; CI warns if constants are bypassed.

- **Record and replay.** Allow replaying recorded video through the pipeline to tune settings without going to the field. *How we check it:* reproduce a field issue offline and show a tuned improvement.
- **Log everything important.** Save timestamps, detections, control outputs, states, and operator actions; bundle logs with the video. *How we check it:* every key requirement maps to at least one metric from the logs.

Appendix — Reflection

1. What went well while writing this deliverable?

   The scope of the work and the scope of the product are well defined, thanks to the detailed explaination from the author of the template. (There is a 80 page pdf explaination can be found on line) In addition, one of the team member have experience being in a rocketry competition, so we are able to understand the work and the working environment.

2. What pain points did you experience during this deliverable, and how did you resolve them?

   The fit criterion is challenging to write, especially for non-functional requirements. For example, there is not a clear way to test if the user interface appears professional and reliable. We resolved this by deferring to the client's judgement. Another pain point is it is hard to divide up the work between the team members. Because all the sections are related to each other, after the team members each finish their own sections, we often need to make large changes to different sections so they are consistent.

3. How many of your requirements were inspired by speaking to your client(s) or their proxies (e.g. your peers, stakeholders, potential users)?

   Because one of the team member already have experience being in a rocketry competition, we are able to come up with most of the requirements by ourselves. Additionally, we have talked to one of the camera operator, but that did not result in any additional requirements.

4. Which of the courses you have taken, or are currently taking, will help your team to be successful with your capstone project.

   - SFWRENG 3XB3: Software Engineering Practice and Experience
   - SFWRENG 3RA3: Software Requirements and Security Considerations

5. What knowledge and skills will the team collectively need to acquire to successfully complete this capstone project? Examples of possible knowledge to acquire include domain specific knowledge from the domain of your application, or software engineering knowledge, mechatronics knowledge or computer science knowledge. Skills may be related to technology, or writing, or presentation, or team management, etc. You should look to identify at least one item for each team member.

   We specifically picked this project because it aligns with our existing skills and domain knowledge. So there is not much new technical knowledge to acquire. For all the domain knowledge required for this project, at least one team member already has the required knowledge. Mike Chen and Xiaotian Lou: domain knowledge of computer vision algorithms. Jianqing Liu: domain knowledge of gimbal and cameras. Zifan Si: domain knowledge of UI frameworks and communication between the frontend and the backend.
   One skill the team collectively need to look into is project and team management skills.

Currently we are having trouble with dividing up the work and staying on track with the timeline.

6. For each of the knowledge areas and skills identified in the previous question, what are at least two approaches to acquiring the knowledge or mastering the skill? Of the identified approaches, which will each team member pursue, and why did they make this choice?

   We don't miss any technical knowledge so I'll talk about how we want to improve our project management in this question. As a team, we need to actually stick to the meeting schedule and better hold ourselves accountable. We need to split tasks into smaller chunks and assign each a short-term deadline, instead of a big task with a deadline that is far in the future.